Motion Planning for Unlabeled Discs with Optimality Guarantees

Kiril Solovey*, Jingjin Yu†, Or Zamir* and Dan Halperin*

***Abstract*—We study the problem of path planning for unlabeled (indistinguishable) unit-disc robots in a planar environment cluttered with polygonal obstacles. We introduce an algorithm which minimizes the total path length, i.e., the sum of lengths of the individual paths. Our algorithm is guaranteed to find a solution if one exists, or report that none exists otherwise. It runs in time $\tilde{O}\left(m^4+m^2n^2\right)$, where m is the number of robots and n is the total complexity of the workspace. Moreover, the total length of the returned solution is at most OPT$+4m$, where OPT is the optimal solution cost. To the best of our knowledge this is the first algorithm for the problem that has such guarantees. The algorithm has been implemented in an exact manner and we present experimental results that attest to its efficiency.**

I. Introduction

We study the problem of path planning for unlabeled (i.e., indistinguishable) unit-disc robots operating in a planar environment cluttered with polygonal obstacles. The problem consists of moving the robots from a set of *start* positions to another set of *goal* positions. Throughout the movement, each robot is required to avoid collisions with obstacles and well as with its fellow robots. Since the robots are unlabeled, we only demand that at the end of the motion each position will be occupied by exactly one robot, but do not insist on a specific assignment of robots to goals (in contrast to the standard labeled setting of the problem).

Following the methodology of performing target assignments and path planning concurrently [1, 2, 3] and adopting a graph-based vertex ordering argument from [4], we develop a complete combinatorial algorithm to the unlabeled problem described above. Our algorithm makes two simplifying assumptions regarding the input. First, it assumes that each start or goal position has a (Euclidean) distance of at least $\sqrt{5}$ to any obstacle in the environment (recall that the robot discs have unit radius) . Additionally, it requires that the distance between each start or goal position to any other such position will be at least 4 (see Figure 1). Given that the two separation conditions are fulfilled, out algorithm is guaranteed to generate a solution if one exists, or report that none exists otherwise. It has a running time[1] of $\tilde{O}\left(m^4+m^2n^2\right)$, where m is the number of robots, and n is the description complexity of the workspace environment, i.e., the number of edges of the polygons. Furthermore, the total length of the returned solution, i.e., the sum of lengths of the individual paths, is at most OPT $+\,4m$, where OPT is the optimal solution cost.

This paper should be juxtaposed with another work by the authors [5] in which we show that a slightly different setting of the unlabeled problem is computationally intractable. In particular, we show that the unlabeled problem of unit-square robots translating amid polygonal obstacles is PSPACE-hard. Our proof relies on a construction of gadgets in which start and goal positions are very close to one another or to the obstacles, i.e., no separation is assumed. This can be viewed as a justification to the assumptions used in the current paper which allow an efficient solution of the problem.

We make several novel contributions. To the best of our knowledge, we are the first to describe a complete algorithm that fully addresses the problem of planning minimum total-distance paths for unlabeled discs. We mention that Turpin et al. [3] describe a complete algorithm for unlabeled discs; however their algorithm minimizes the maximal path length of an individual robot. Furthermore, our algorithm makes more natural assumptions on the input problem than the ones made by Turpin et al. We also mention the work by Adler et al. [6] in which a complete algorithm is described, but it does not guarantee optimality. While the latter work makes the same assumption as we make on a separation of 4 between starts and goals, it does not assume separation from obstacles. However, it requires that the workspace environment will consist of a simple polygon. On the practical side, we provide an exact and efficient implementation which does not rely on non-deterministic procedures, unlike sampling-based algorithms. The implementation will be made publicly available upon the publication of this paper.

Our work is motivated by theoretical curiosity as well as potential real world applications. From a theoretical standpoint, we have seen great renewed interests in developing planning algorithms for multi-robot systems in continuous domains (see, e.g., [3, 7, 8, 9, 10]) and discrete settings (see, e.g., [2, 11]). Whereas significant headway has been made in solving multi-robot motion-planning problems, many challenges persist; the problem studied in this paper—path planning for unlabeled disc robots in a general environment with guarantees on total distance optimality—remains unresolved (until now). On the

*K. Solovey, O. Zamir and D. Halperin are with the Blavatnic School of Computer Science, Tel Aviv University, Israel; Email: {kirilsol,danha}@post.tau.ac.il, orzamir@mail.tau.ac.il; The work of K.S. and D.H. has been supported in part by the Israel Science Foundation (grant no. 1102/11), by the German-Israeli Foundation (grant no. 1150-82.6/2011), and by the Hermann Minkowski-Minerva Center for Geometry at Tel Aviv University.

†J. Yu is with the Computer Science and Artificial Intelligence Lab at the Massachusetts Institute of Technology; Email: jingjin@csail.mit.edu; His work has been supported in part by ONR projects N00014-12-1-1000 and N00014-09-1-1051.

[1]For simplicity of presentation, we omit log factors when stating the complexity of the algorithm, and hence use the $\tilde{O}$ notation.

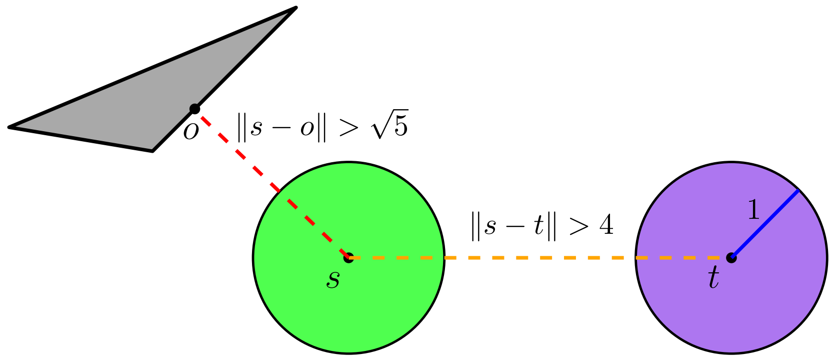


Fig. 1: Illustration of the separation conditions assumed in this work. The green and purple discs represent two unit-disc robots placed in $s \in S, t \in T$, respectively. The blue line represents the unit radius of the robot (for scale). The distance between s and t is at least 4 units (see dashed orange line). The gray rectangle represents an obstacle, and the point o represents the closest obstacle point to s. Notice that the distance between o and s is at least $\sqrt{5}$ units (see dashed red line).

application side, in the past few years, we have witnessed the rapid development and adaptation of autonomous multi-robot and multi-vehicle systems in a wide variety of application domains. The most prominent example is arguably the success of Kiva Systems, now part of Amazon, which developed a warehousing system employing hundreds of mobile robots to facilitate order assembly tasks [12]. More recently, Amazon, DHL, and Google have demonstrated working prototypes of aerial vehicles capable of automated package delivery. Since the vehicles are intended to operate in an autonomous, swarm-like setting, we can foresee in the near future the emerging demand of efficient path planning algorithms designed for such systems. We note that when the warehouse robots or the delivery aerial vehicles do not carry loads, they are effectively unlabeled robots. In such scenarios, planning collision-free, total-distance near-optimal paths translates into allowing the vehicles, as a whole, to travel with minimum energy consumption.

The rest of the paper is organized as follows. In Section II we review related work. In Section III we provide an overview of our algorithm and necessary background material. In Section IV we establish several basic properties of the problem and describe the algorithm in Section V. We report on experimental results in Section VI and conclude with a discussion in Section VII.

II. Related work

The problem of multi-robot motion planning is notoriously challenging as it often involves many degrees of freedom, and consequently a vast search space [13, 14]. In general, each additional robot introduces several more degrees of freedom to the problem. Nevertheless, there is a rich body of work dedicated to this problem. The earliest research efforts can be traced back to the 1980s [15].

Approaches for solving the problem can be typically subdivided into categories. Decoupled techniques (see, e.g., [16, 17, 18, 19, 20, 21]) reduce the size of the search space by partitioning the problem into several subproblems, which are solved separately, and then the different components are combined. In contrast to that, centralized approaches (see, e.g., [22, 23, 24, 25, 26, 27, 28]) usually work in the combined high-dimensional configuration space, and thus tend to be slower than decoupled techniques. However, centralized algorithms often come with stronger theoretical guarantees, such as completeness. Besides these, the multi-robot motion-planning problem has also been attacked using methods based on network flow [8] and mixed integer programming [29], among others.

Multi-robot motion planning can also be considered as a discrete problem on a graph [30]. In this case the robots are pebbles placed on the vertices of the graph and are bound to move from one set of vertices to another along edges. Many aspects of the discrete case are well understood. In particular, for the labeled setting of the problem there exist efficient feasibility-test algorithms [31, 32, 33], as well as complete planners ([9, 34, 35]). For the unlabeled case, there even exist complete and efficient planners that generate the optimal solution [4, 11, 36] under different metrics. While there exists a fundamental difference between the discrete and the continuous setting of the multi-robot problem, the continuous case being exceedingly more difficult, several recent techniques in the continuous domain [3, 6, 37] have employed concepts that were initially introduced in the discrete domain.

As mentioned above, the works by Adler et al. [6] and Turpin et al. [3] solve very similar settings of the unlabeled problem for disc robots to the one treated in this paper, only with different assumptions and goal functions, and thus it is important to elaborate on these two techniques. Adler et al. [6] show that the unlabeled problem in the continuous domain can be transformed into a discrete *pebble-motion on graph* problem. Their construction guarantees that in case a solution to the former exists, it can be generated by solving a discrete pebble problem and adapted to the continuous domain. In particular, a motion of a pebble along an edge is transformed into a motion of a robot along a local path in the free space. Turpin et al. [3] find an assignment between starts and goals which minimize the longest path length traveled by any of the robots. Given such an assignment a shortest path between every start position to its assigned goal is generated. However, such paths still may result in collisions between the robots.

The authors show that collisions can be elegantly avoided by prioritizing the paths. Our current work follows to some extent a similar approach, although in our case some of the robots must slightly stray from the precomputed paths in order to guarantee completeness—a thing which makes our technique much more involved. This follows from the fact that we make milder assumptions on the input. Another difference is that the goal function of our algorithm is to minimize the total path length, which requires very different machinery than the one used by Turpin et al.

III. Preliminaries and algorithm overview

Our problem consists of moving m indistinguishable unit-disc robots in a workspace $\mathcal{W} \subset \mathbb{R}^2$ cluttered with polygonal obstacles, whose overall number of edges is n. We define $\mathcal{O} := \mathbb{R}^2 \setminus \mathcal{W}$ to be the complement of the workspace, and we call $\mathcal{O}$ the *obstacle space*. For given $r \in \mathbb{R}_+, x \in \mathbb{R}^2$, we define $\mathcal{B}_r(x)$ to be the open disc of radius r, centered at x. For given $r \in \mathbb{R}_+, X \subset \mathbb{R}^2$ we also define $\mathcal{B}_r(X) := \bigcup_{x \in X} \mathcal{B}_r(x)$.

We consider the unit-disc robots to be open sets. Thus a robot avoids collision with the obstacle space if and only if its center is at distance at least 1 from $\mathcal{O}$. More formally, the collection of all collision free configurations, termed the *free space*, can be expressed as $\mathcal{F} := \left\{x \in \mathbb{R}^2 : \mathcal{B}_1(x) \cap \mathcal{O} = \emptyset\right\}$. We also require the robots to avoid collisions with each other. Thus for every given two configurations $x, x' \in \mathcal{F}$ two distinct robots can be placed in x and x' only if $\|x - x'\| \geqslant 2$. Throughout this paper the notation $\|\cdot\|$ will indicate the L_2 norm.

In addition to the workspace $\mathcal{W}$ we are given sets $S = \{s_1, s_2, ..., s_m\}$ and $T = \{t_1, t_2, ..., t_m\}$ such that $S, T \subset \mathcal{F}$. These are respectively the sets of *start* and *goal* configurations of our m indistinguishable disc robots. The problem is now to plan a collision-free motion for m unit-disc robots such that each of them starts at a configuration in S and ends at a configuration in T. Since the robots are indistinguishable or *unlabeled*, it does not matter which robot ends up at which goal configuration, as longs as all the goal configurations are occupied at the end of the motion. Formally, we wish to find m paths $\pi_i : [0,1] \to \mathcal{F}$, for $1 \leqslant i \leqslant m$, such that $\pi_i(0) = s_i$ and $\bigcup_{i=1}^{m} \pi_i(1) = T$. Furthermore, we are interested in finding a set of such paths which minimizes the expression $\sum_{i=1}^{m} |\pi_i|$, where $|\pi_i|$ represents the length of π_i in the L_2 norm.

A. Simplifying assumptions

By making the following simplifying assumptions (see Figure 1) we are able to show that our algorithm is complete and near-optimal. The first assumption that we make is identical to the one used by Adler et al. [6]. It requires that every pair of start or goal positions will be separated by a distance of at least 4:

$$\forall v, v' \in S \cup T, \quad \|v - v'\| \geqslant 4. \tag{1}$$

The motivation for the above assumption is the ability to prove the existence of a *standalone goal*—a goal position that does not block other paths, assuming that the paths minimize the total length of the solution.

We also need the following assumption in order to guarantee that the robots will be able switch targets, in case that a given assignment of robots to goals induces collision between the robots:

$$\forall v \in S \cup T \text{ and } \forall x \in \mathcal{O}, \quad \|v - x\| \geqslant \sqrt{5}. \tag{2}$$

B. Overview of algorithm

Here we provide an overview of our technique. Recall that our problem consists of S, T, which specify the set of start and goal positions for a collection of m unit-disc robots, and a workspace environment $\mathcal{W}$.

We describe the first iteration of our recursive algorithm. For every $s_i \in S, t_j \in T$ we find the shortest path $\gamma_i^j : [0,1] \to \mathcal{F}$ from s_i to t_j. Among these m^2 paths we select a set of m paths $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$, where $\gamma_i : [0,1] \to \mathcal{F}$ for every $1 \leqslant i \leqslant m$, such that $\bigcup_{i=1}^{m} \{\gamma_i(0)\} = S, \bigcup_{i=1}^{m} \{\gamma_i(1)\} = T$. Furthermore, we require that Γ will be the path set that minimizes the total length of its paths under these conditions. Note that at this point we only require that the robots will not collide with obstacles, and do not worry about collisions between the robots. The generation of Γ is described in detail in Section V, Theorem 13.

In the next step we find a goal $t \in T$ which does not block paths in Γ that do not lead to t. We call such a t a *standalone goal*. Next, we find a start $s \in S$ from which a robot will be able to move to t without colliding with other robots situated in the rest of the start positions. We carefully select such a start s and generate the respective path to t in order to minimize the cost of the returned solution. We prepare the input to the next iteration of the algorithm by assigning $S := S \setminus \{s\}, T := T \setminus \{t\}$, and by treating the robot placed in t as an obstacle, i.e., $\mathcal{W} := \mathcal{W} \setminus \mathcal{B}_1(t)$.

IV. Theoretical foundations

In this section we establish several basic properties of the problem. Recall that our problem is defined for a workspace $\mathcal{W}$, whose free space for a single unit-disc robot is $\mathcal{F}$. Additionally, we have two sets of start and goal positions $S = \{s_1, \ldots, s_m\}, T = \{t_1, \ldots, t_m\}$, respectively.

The following lemma implies that if a robot moving from some start position $s \in S$ along a given path collides with a region $\mathcal{B}_1(t)$, for some other goal $t \in T$, then there exists another path $\gamma \in \mathcal{F}$ which moves the robot from s to t. In the context of our algorithm this lemma implies that if a path for a robot interferes with some other goal position, then the path can be modified such that it will move the robot to the interfering goal instead.

Lemma 1. *Let $v \in S \cup T$ and $x \in \mathcal{F}$ such that $\mathcal{B}_1(x) \cap \mathcal{B}_1(v) \neq \emptyset$. Then the straight-line path from x to v is contained in $\mathcal{F}$, i.e., $\overline{xv} \subset \mathcal{F}$.*

Proof: Here we use the fact that for every $o \in \mathcal{O}$ it holds that $\|v - o\| \geqslant \sqrt{5}$, which is the second separation assumption. Without loss of generality, assume that the straight-line segment from v to x is parallel to the x-axis. Denote by A and B the bottom points of the unit discs around v and x, respectively (see Figure 2). Similarly, denote by C and D the top points of these discs. By definition of v, x, we know that $\|v-x\|< 2$. Thus, $\|v-B\|=\|v-D\|<\sqrt{5}$ (see dashed red segment). This implies that the rectangle defined by the points A, B, C, D is entirely contained in $\mathcal{W}$ (see orange square). Thus, the straight-line path $\overline{xv}$ is fully contained in $\mathcal{F}$. ■

Definition 2. *Let Γ be a set of m paths $\{\gamma_1, \ldots, \gamma_m\}$ such that for every $1 \leqslant i \leqslant m$, $\gamma_i : [0,1] \rightarrow \mathcal{F}$, $S = \bigcup_{i=1}^{m}\{\gamma_i(0)\}$, $T = \bigcup_{i=1}^{m}\{\gamma_i(1)\}$. We call Γ the* optimal-assignment path set *for $S, T, \mathcal{F}$, if it minimizes the expression $|\Gamma|:=\sum_{i=1}^{m}|\gamma_i|$, over all such path sets.*

Note that Γ is not necessarily a feasible solution to our problem since at this stage we still ignore possible collisions between robots. Let $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ be an optimal-assignment path set for $S, T, \mathcal{F}$. Without loss of generality, assume that for every $1 \leqslant i \leqslant m$, $\gamma_i(0) = s_i, \gamma_i(1) = t_i$.

Definition 3. *Given an optimal-assignment path set Γ, we call $t_k \in T$ a* standalone goal, *for some $1 \leqslant k \leqslant m$, if for every $\gamma_i \in \Gamma, i \neq k$ it holds that $\mathcal{B}_1(t_k) \cap \mathcal{B}_1(\gamma_i) = \emptyset$.*

Standalone goals play a crucial role in our algorithm. We first show that at least one such goal always exists.

Theorem 4. *Let Γ be an optimal-assignment path set. Then there exists a standalone goal.*

Proof: Assume by contradiction that every goal t_i interferes with some path $\gamma_j \in \Gamma, i \neq j$. This implies that there is a circular interference, i.e., there exist $\ell \leqslant m$ goals, which we denote, for simplicity and without loss of generality, as $t_1, \ldots, t_\ell$ such that for every $1 < i \leqslant \ell$, $\mathcal{B}_1(t_i) \cap \mathcal{B}_1(\gamma_{i-1}) \neq \emptyset$, and $\mathcal{B}_1(t_1) \cap \mathcal{B}_1(\gamma_\ell) \neq \emptyset$. More formally, let $\mathcal{I}$ be a directed graph vertices are T. For every t_i which interferes with a path γ_j, where $j \neq i$ we draw an edge from t_i to t_j. If there exists no standalone goal then $\mathcal{I}$ has a directed cycle of size greater than one. This is due to the fact that if $\mathcal{I}$ were *directed acyclic* then it should have had a node whose *out degree* is zero.

We show that in this case, the paths $\gamma_1, \ldots, \gamma_\ell$ do not induce the minimal assignment for the starts and goals $\{s_1, \ldots, s_\ell\}, \{t_1, \ldots, t_\ell\}$. This would imply that Γ is not the optimal assignment path set. We claim that instead of assigning s_i to t_i we may assign s_i to t_{i+1} for $1 \leqslant i < \ell$ and from s_ℓ to t_1 and get a collection of paths $\gamma'_1, \ldots, \gamma'_\ell$, such that $|\gamma'_i|< |\gamma_i|$ for every $1 \leqslant i \leqslant \ell$. Denote by $x \in \gamma_i$ the first interference point with t_{i+1} along γ_i. Additionally, denote by γ_i^x the subpath of γ_i that starts with s_i and ends with x. Define γ'_i to be the concatenation of γ_i^x and xt_{i+1}.

We need to show first that $\gamma'_i \subset \mathcal{F}$ and that $|\gamma'_i|< |\gamma_i|$. The subpath γ_i^x is obstacle-collision free, and so is $\overline{xt_{i+1}}$ according to Lemma 1. Thus, γ'_i is free as well. Now, note that $\|x - t_{i+1}\|< 2$ and $\|t_{i+1} - t_i\| \geqslant 4$. Thus, by the triangle inequality $\|x - t_i\| \geqslant 2$. This finishes our proof. ■

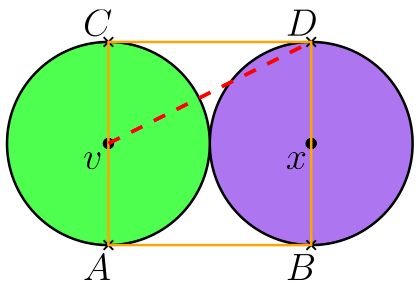


Fig. 2: Illustration of the proof of Lemma 1.

We introduce the notion of 0-*hop* and 1-*hop* paths. Informally, a 0-hop path is a path assigned to the standalone goal which is not blocked by any start position.

Definition 5. *Let t_k be a standalone goal. The path γ_k is called a 0-hop path if for every $s_i \in S, i \neq k$ it follows that $\mathcal{B}_1(s_i) \cap \mathcal{B}_1(\gamma_k) = \emptyset$.*

Definition 6. *Let t_k be a standalone goal, and suppose that γ_k is not 0-hop. Let $x \in \gamma_k$ be the farthest point along γ_k for which there exists $s_i \in S$ such that $\mathcal{B}_1(s_i) \cap \mathcal{B}_1(x) \neq \emptyset$. The 1-hop path, which is denoted by H_i^k, is a concatenation of the straight-line path $\overline{s_i x}$ and a subpath of γ_k that starts at x and ends in t_k.*

Namely, the 1-hop path moves the robot situated in s_i, which interferes with γ_k, to t_k (see Figure 3). The following theorem shows that a robot situated in s_i that blocks the 0-hop path can be moved to t_k without inducing collisions with other robots.

Theorem 7. *Let t_k be a standalone goal. Suppose that γ_k is not 0-hop and let H_i^k be the 1-hop path from s_i to t_k. Then it holds that for every $s_j \in S, j \neq i, k$, $\mathcal{B}_1(H_i^k) \cap \mathcal{B}_1(s_j) = \emptyset$.*

Proof: By separation of start positions, and by definition of x (see Definition 6), there exists a single start position s_i for which $\mathcal{B}_1(x) \cap \mathcal{B}_1(s_i) \neq \emptyset$. By Lemma 1, and the separation condition, the path $\overline{s_i x}$, does not induce collisions between a robot moving along it and other robots placed in s_j, for $1 \leqslant j \leqslant m, j \neq i, k$. Finally, by definition of x, the rest of the path H_i^k, which is a subpath of γ_k, is free of collisions with a robot situated in s_j. ■

After moving a robot from s_i to t_k we have to ensure that some other robot can reach t_i. We define the following *switch* path (see Figure 3).

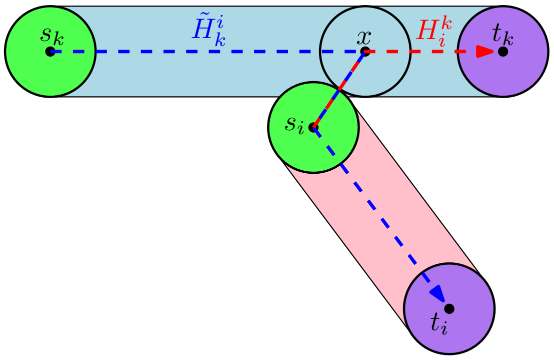


Fig. 3: Example of 1-hop and switch paths. t_k is a standalone goal. The 1-hop path H_i^k is drawn as a dashed red curve, while the switch path $\tilde{H}_k^i$ is drawn as a dashed blue curve.

Definition 8. *Let t_k be a standalone goal and suppose that γ_k is not 0-hop. The* switch path *$\tilde{H}_k^i$ is a concatenation of the following paths: (1) the subpath of γ_k that starts in s_k and ends in x; (2) $\overline{xs_i}$; (3) γ_i.*

Lemma 9. *Let $\tilde{H}_i^k$ be a switch path. Then $\mathcal{B}_1(\tilde{H}_i^k) \cap \mathcal{B}_1(t_k) = \emptyset$.*

Proof: $\tilde{H}_i^k$ can potentially interfere with t_k only along $\overline{xs_i}$, due to the definition of x. For any $y \in \overline{xs_i}$ it follows that $\|y - s_i\| < 2$. If $\|y - t_k\| < 2$ then it follows that $\|s_i - t_k\| < 4$, which is a contradiction. ■

Corollary 10. *Let $H_i^k, \tilde{H}_k^i$ be the 1-hop and switch paths, respectively. Then $|H_i^k| + |\tilde{H}_k^i| = |\gamma_i| + |\gamma_k| + 4$.*

V. Near-optimal algorithm for unlabeled planning

In this section we describe our algorithm for the unlabeled multi-robot motion planning problem of unit-disc robots moving amid polygonal obstacle and establish its completeness. Additionally, we bound the cost of the returned solution. Finally, we analyze the complexity of the algorithm.

A. The algorithm

We describe our recursive algorithm, which returns a set of m paths Π. Recall that the input consists of S, T and a workspace $\mathcal{W}$, which induces the free space $\mathcal{F}$. The algorithm first produces the optimal-assignment path set Γ. Let OPT be the optimal solution cost and note that $|\Gamma| \leqslant$ OPT since $|\Gamma|$ is a lower bound on the actual cost, as interactions between the robots might increase the traversed distance.

Let t_k be a standalone goal, which exists according to Theorem 4. Suppose that the 0-hop path γ_k is not blocked by any other robot located in a start position. Then, γ_k is added to Π and the algorithm is run recursively on the input $S' := S \setminus \{s_k\}, T' := T \setminus \{t_k\}$, with the workspace $\mathcal{W}' := \mathcal{W} \setminus \mathcal{B}_1(t_k)$, which results in the free space $\mathcal{F}' := \mathcal{F} \setminus \mathcal{B}_2(t_k)$.

Alternatively, in case that γ_k is blocked, i.e., in interference with some $s_i \in S, i \neq k$, the algorithm produces the 1-hop path H_i^k, as described in Theorem 7, and adds it to Π. Then, the algorithm is run recursively on the input $S' := S \setminus \{s_i\}, T' := T \setminus \{t_k\}$, with the workspace $\mathcal{W}' := \mathcal{W} \setminus \mathcal{B}_1(t_k)$, and the free space $\mathcal{F}' := \mathcal{F} \setminus \mathcal{B}_2(t_k)$.

B. Completeness and near-optimality

We first show that the algorithm is guaranteed to find a solution, if one exists, or report that none exists otherwise.

Theorem 11. *Given an input $S, T, \mathcal{W}$, which complies with assumptions 1 ,2 (Section III-A), and for which the number of start and goal positions for every connected component of $\mathcal{F}$ is the same, the algorithm is guaranteed to find a solution for the unlabeled multi-robot motion-planning problem.*

Proof: Consider the first iteration of the algorithm. Let $\Gamma := \{\gamma_1, \ldots, \gamma_m\}$ be the optimal-assignment path set and let t_k be a standalone goal.

Suppose that γ_k is a 0-hop path. Then, for every $j \neq k$, $\gamma_j \subset \mathcal{F} \setminus \mathcal{B}_2(t_k)$, since t_k is a standalone goal. Now suppose that γ_k is not a 0-hop path. By Theorem 7 the 1-hop path H_i^k from s_i to t_k does not collide with any other start position. By Lemma 9 $\tilde{H}_k^i \subset \mathcal{F} \setminus \mathcal{B}_2(t_k)$. Additionally, notice that for every $j \neq i, k$, $\gamma_j \subset \mathcal{F} \setminus \mathcal{B}_2(t_k)$.

Thus, in any of the two situations, the removal of the standalone goal does not separate between start and goal configurations that are in the same connected component of $\mathcal{F}$. Note that in the first level of the recursion the existence of a standalone goal t_k guarantees the existence of a path to t_k which does not collide with the other robots. This is possible due to assumptions 1, 2. Thus, in order to ensure the success of the following recursions, we need to show that these assumptions are not violated. First, note that assumption 1 is always maintained, since we do not move existing start and goal positions. Secondly, when a robot situated in t_k is treated as an obstacle added to the set $\mathcal{O} := \mathcal{O} \cup \mathcal{B}_1(t_k)$ (or conversely removed from $\mathcal{F}$, as described above), the first assumption induces the second. ■

We proceed to prove the near-optimality of our solution.

Theorem 12. *Let Π be the solution returned by our algorithm and let Γ be the optimal-assignment path set for $S, T, \mathcal{W}$. Then $|\Pi| \leqslant$ OPT $+ 4m$, where OPT is the optimal solution cost.*

Proof: Let $\Gamma := \{\gamma_1, \ldots, \gamma_m\}$ be the optimal-assignment path set for the input $S, T, \mathcal{W}$, and let t_k be a standalone. Additionally, assume that the algorithm generated the 1-hop path H_i^k, since γ_k was blocked (the case when γ_k is not blocked is simpler to analyze). Let $\tilde{H}_k^i$ be the switch path from s_k to t_i. Similarly, denote by Γ' the optimal-assignment path set for $S' := S \setminus \{s_i\}, T' := T \setminus \{t_k\}$ and the workspace $\mathcal{W}' := \mathcal{W} \setminus \mathcal{B}_1(t_k)$, which induces the free space $\mathcal{F}'$.

We will show that $|\Gamma'| + |H_i^k| \leqslant \text{OPT} + 4$. As mentioned in Theorem 11, for every $j \neq i, k$, it follows that for $\gamma_j \in \Gamma$ and $\gamma_j \subset \mathcal{F}'$. We also showed that the same holds for $\tilde{H}_k^i$, i.e., $\tilde{H}_k^i \subset \mathcal{F}'$. Thus, the path set $R := (\Gamma \setminus \{\gamma_i, \gamma_k\}) \cup \{\tilde{H}_k^i\}$ represents a valid assignment for $S', T', \mathcal{F}'$, even though it might not be optimal. This means that $|\Gamma'| \leqslant |R|$. Thus,

$$|\Gamma'| + |H_i^k| \leqslant |R| + |H_i^k| = \sum_{j \neq i,k}^{m} |\gamma_j| + |\tilde{H}_k^i| + |H_i^k|$$
$$\leqslant \sum_{k=1}^{m} |\gamma_k| + 4 = |\Gamma| + 4 \leqslant \text{OPT} + 4,$$

where the third step is due to Corollary 10. Thus, every level of the recursion introduces and additive error factor of 4. Repeating this process for the m iterations we obtain $|\Pi| \leqslant \text{OPT} + 4m$. ■

C. Complexity Analysis

We analyze the complexity of the algorithm. In order to do so, we have to carefully consider the operations that are performed in every iteration of the algorithm.

Theorem 13. *Given m unlabeled unit-disc robots operating in a polygonal workspace with n vertices, the algorithm described* *above returns a solution, or reports that none exists otherwise, with a running time of* $\tilde{O}\left(m^4 + m^2n^2\right)$.

Proof: Let us consider a specific iteration j. The input of this iteration consists of $m-j+1$ start positions S_j and $m-j+1$ goal positions T_j. The workspace region of this iteration is defined to be $\mathcal{W}_j := \mathcal{W} \setminus (\mathcal{B}_1(T \setminus T_j))$, which induces the free space $\mathcal{F}_j$. Note that $S_1 \equiv S, T_1 \equiv T, \mathcal{F}_1 \equiv \mathcal{F}, \Gamma_1 \equiv \Gamma$.

In order to find the optimal-assignment path set Γ_j for $S_j, T_j, \mathcal{F}_j$ one has to first find the shortest path in $\mathcal{F}_j$ for every pair of start and goal positions $s \in S_j, t \in T_j$. Given the costs of all those combinations, the Hungarian algorithm [38][2] finds the optimal assignment, and so Γ_j is produced. In the next step, a standalone goal t_k is identified and it is checked whether the 0-hop path leading to t_k is clear, in which case γ_k is added to Π. If it is not, one needs to find the last start position that interferes with this path and generate the respective 1-hop path, which will be included in Π.

The complexity of finding a shortest path for a disc depends on the complexity of the workspace, which in our case is $O(n+j)$. This task is equivalent to finding a shortest path for a point robot in $\mathcal{F}_j$. The generation of $\mathcal{F}_j$ can be done in $O\left((n+j)\log^2(n+j)\right)$ time complexity, and the overall complexity of this structure would be $O(m+j)$ [39]. A common approach for finding shortest paths in the plane is to construct a *visibility graph* [40] which encapsulates information between every pair of vertices of a given arrangement of segments, while avoiding crossings with the segments. In our case, the arrangement should include $\mathcal{F}_j$ as well as all the points from S_j and T_j. Thus, we would need to generate a visibility graph over a generalized polygon of total complexity $O(m+n)$. This can be done in $\tilde{O}\left((m+n)^2\right) = \tilde{O}(m^2+n^2)$ [40][3]. Given the visibility graph with $O(m+n)$ vertices and $O(m^2+n^2)$ edges we find for each $s \in S_j$ the shortest path to every $t \in T_j$. For each s we run the Dijkstra algorithm which requires $O(m^2+n^2)$ time, and finds the shortest path from the given s to any $t \in T_j$. Since $|S_j| = m-j+1$, the total running time of finding the shortest path from every start to every goal is $O((m-j)(m^2+n^2))$.

To find Γ_j we employ the Hungarian algorithm [38], which runs in $O\left((m-j)^3\right)$ time. Now, given Γ_j we wish to find a standalone goal t_k. We first note that the complexity of each path in Γ_j is bounded by the complexity of $\mathcal{F}_j$, which is $O(n+j)$. For every $t_i \in T_j, \gamma_{i'} \in \Gamma_j, i \neq i'$ we check whether $\mathcal{B}_1(t_i) \cap \mathcal{B}_1(\gamma_{i'}) \neq \emptyset$. This step has a running time of $\tilde{O}((m-j)(m-j)(n+j)) = \tilde{O}((m-j)^2(n+j))$. Finding the last closest blocking start from S_j of the path γ_k takes additional $O((m-j)(n+j))$ time by going over all starts in S_j and comparing the distance of their interference point with γ_k. Thus, the overall complexity of a given iteration j is

$$\tilde{O}\left((m-j)(m^2+n^2)+(m-j)^3+(m-j)^2(n+j)+(m-j)^2\right)$$
$$= \tilde{O}\left((m-j)(m^2+n^2)\right).$$

Note that the cost of the different components is absorbed in the cost of calculating the shortest paths. We conclude with the running time of the entire algorithm:

$$\tilde{O}\left(\sum_{j=1}^{m}(m-j)(m^2+n^2)\right) = \tilde{O}\left(m^4+m^2n^2\right).$$

■

VI. Experimental results

We implemented the algorithm and evaluated its performance on various challenging scenarios.

A. Implementation details

First, we wish to emphasize that our implementation deals with geometric primitives, e.g., polygons and discs, and does not rely on any graph discretization of the problem. The implementation relies on exact geometric methods that are provided by CGAL [41]. As such, it is complete, robust and deterministic. In addition to that, the implementation is parameter-free.

We implemented the algorithm in C++ and relied heavily on CGAL for geometric computing, and in particular on the `Arrangement_2` package [42]. Generation of the free space was done using *Minkowski sums*, while shortest paths were generated using *visibility graphs* [40]. In order to find the optimal assignment, we used a C++ implementation of the *Hungarian algorithm* [38], available at [43].

B. Test scenarios

We report in Table I on the running time of the algorithm for the four scenarios depicted in Figure 4. The grid scenario (Figure 4a) is used to illustrate the performance of the algorithm in a sterile obstacles-free workspace. The triangles and cross scenarios (Figure 4b,4d) include multiple obstacles, which have a tremendous affect on the performance of the algorithm. The maze scenario (Figure 4c), which also includes multiple obstacles, has several narrow passages.

It is evident that the running time is dominated by shortest-path calculations (see Theorem 13). The second largest contributor to the running time of the algorithm, is the calculation and maintenance of the configuration space, which includes the update of the free space for every iteration. A interesting relation is found between the overall complexity of finding the optimal assignment ($O(m^4)$) and its modest contribution to the actual running time in practice, when compared to the other components. This can be explained by the fact that the implementation of the Hungarian algorithm [43] employs floating-point arithmetic, while the geometric operations, e.g., shortest pathfinding, rely on exact geometric kernels, which have unlimited precision [44]. It is noteworthy that the algorithm produces solutions whose cost is very close to the optimal cost.

[2] Also known as the Kuhn-Munkres algorithm.

[3] We note that in our setting the arrangement consists not only of straight-line segments, but also of circular arcs which are induced by robots situated in target positions. Yet, the algorithm constructing the visibility graph can treat such cases as well, while still guaranteeing the (near-)quadratic run-time complexity mentioned above.

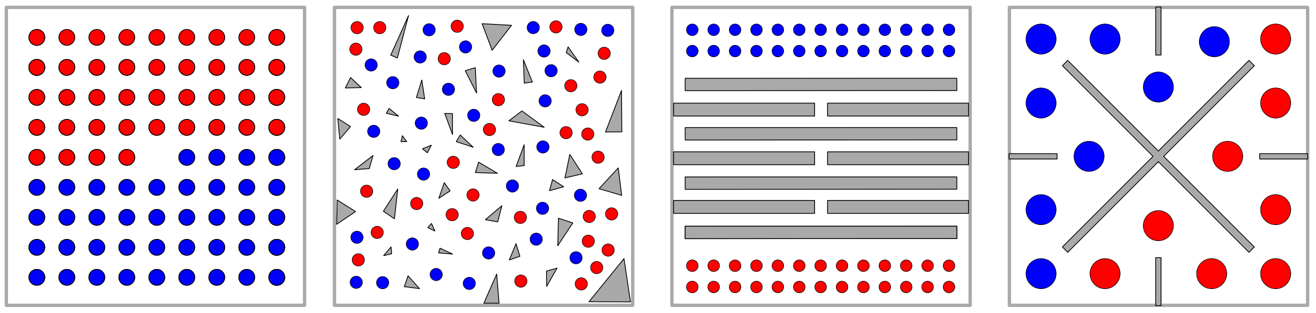

(a) grid (b) triangles (c) maze (d) cross

Fig. 4: Test scenarios. Obstacles are represented as gray polygons. Discs represent robots placed in start (red) and goal (blue) positions. (a) Grid scenario with 40 robots. (b) Triangles scenario with 32 robots and multiple triangular obstacles. (c) Maze scenario with with 26 robots; robots need to pass through a collection of narrow passages in order to reach their goals. (d) Cross scenario with 8 robots and several wall obstacles.

To gather a better understanding of the running time of the algorithm, we also report on the running time of each iteration for the triangles scenario (see Figure 5a). For every iteration $1 \leqslant j \leqslant 32$ we report on the running time of maintaining the configuration space, and finding shortest paths.

Another important aspect of the algorithm is the relation between the number of robots and the performance. To test this relation we performed the following experiment, which is based on the triangles scenario as well. We report in Figure 5b how the performance is affected by an increase or decrease in the number of robots. In order to maintain a similar level of density between the various tests we increase the radius of the robots to the maximal allowed radius which abides by the separation assumptions.

The algorithm and its current implementation can deal with rather complex scenarios. However, it is clear that there is a limit to the number of robots or workspace complexity with which it can cope, due the relatively high degree of the polynomial, in the running time complexity.

VII. Discussion and future work

A. *Relaxing the separation conditions*

The algorithm presented in this paper requires that two conditions will hold. First it is assumed that every pair of start or goal position will be separated by a distance of at least 4. The second condition requires that every start or goal positions will be separated from an obstacle by a distance of at least $\sqrt{5} \approx 2.236$. We believe that the obstacles-separation factor can be lowered to $\sqrt{13 - 6\sqrt{3}} \approx 1.614$ using a tighter mathematical analysis. While the proof for Lemma 1 would change, the rest of the proofs will not require any special modification. In the near future we aim to consider less strict separation requirements such as reducing the robots separation to 3 and removing the requirement that start positions will be separated from goal positions. However, it seems that the machinery described here will not suffice for the tighter setting and different tools will be required.

We note that without adequate obstacle clearance, it is in fact impossible for any algorithm to guarantee a maximum $O(m)$ deviation from the cost of an optimal-assignment path set, regardless of the separation between start and goals. For instance, see the gadget in Figure 6. The two pairs of (blue) start and (red) goal positions satisfy the 4 separation requirement. The disc starting at the bottom left, which we call disc 1, is matched by the optimal assignment with the goal on the top left of the figure, and the disc starting in the middle, which we call disc 2, is matched with the goal on the right. This is due to the fact that disc 1 and disc 2 reside in two disjoint components of the free space. For disc 1 to go through the lower tunnel, disc 2 must move non-trivially (e.g., to the location of the green discs). Let this nontrivial distance be δ. Then, as disc 1 passes through the upper tunnel, another nontrivial move of distance δ must be performed by disc 2 to clear the way. Now, we remove disc 1 from the gadget and stack $m/2$ of the resulting gadget vertically. If we require the rest $m/2$ discs to pass through the stacked construction, a total distance penalty of $\Omega(m^2\delta)$ is incurred.

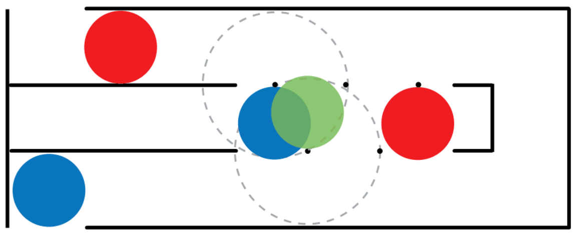
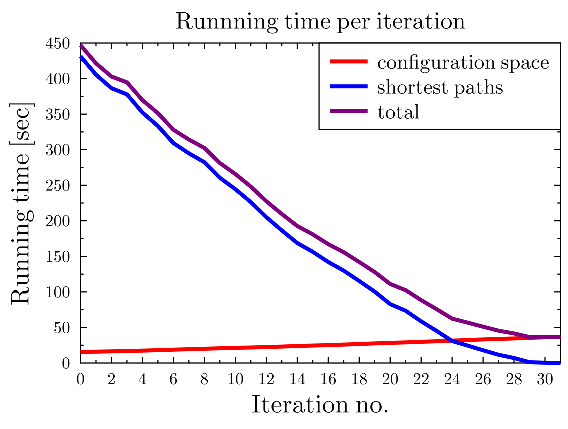


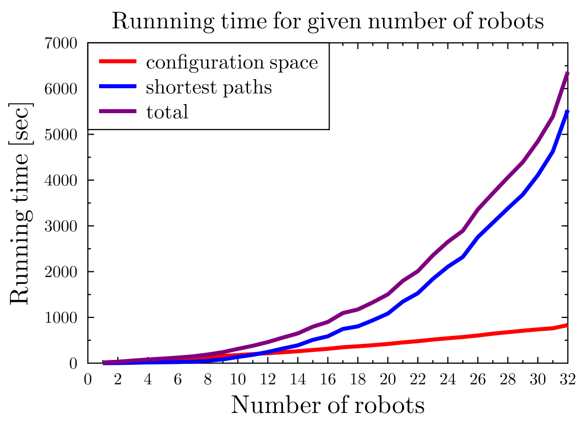
Fig. 6: Illustration of a scenario in which every solution has a deviation of $\Omega(m^2)$ from the cost of the optimal-assignment path set.

B. *Improving the algorithm*

It is evident from our experiments that most of the running time of the algorithm in practice is devoted to finding shortest paths. To be specific, in each iteration the algorithm performs a shortest-path search between every pair of start and target positions. This entire process is performed only for the sake of finding a single standalone goal. It will be interesting to investigate whether more information can be extracted from the optimal assignment returned by the Hungarian algorithm.



(a)



(b)

Fig. 5: Graphs depicting the running time of maintaining the configuration space, and finding shortest path, for the triangles scenario (Figure 4b). (a) Running time is reported separately for every iteration of the algorithm. (b) The behavior of the algorithm for a growing number of robots in the triangles scenario is depicted.

		Scenarios			
		Grid	Triangles	Maze	Cross
scenario properties	robots	40	32	26	8
	workspace vertices	4	118	56	44
	robot radius	0.05	0.034	0.035	0.09
running time (sec.)	configuration space	261	832	141	17
	shortest paths	50	5532	658	1
	optimal assignment	0.016	0.001	0.016	0
	standalone goal	0.001	0.016	0.119	0
	total	**311**	**6394**	**800**	**19**
cost	lower bound	36.33	10.41	154.86	12.11
	actual cost	37.16	10.69	155.21	12.4
algorithm's behavior	0-hop paths	25	32	14	7
	1-hop paths	1	0	12	1

TABLE I: Results of our algorithm for the scenarios depicted in Figure 4. We first describe the properties of each scenario, which consist of the number of robots, the complexity of the workspace (n), and the robot's radius (every scenario is bounded by the $[-1,1]^2$ square). Then, we report the running time (in seconds) of the different components of the algorithm: construction and maintenance of the configuration space; calculation of shortest paths; calculation of the optimal assignment; search for a standalone goal. Then, we report the lower-bound cost of the solution and the cost of the actual solution returned by the algorithm. Finally, we report the number of iterations for which the algorithm returned a 0-hop or a 1-hop path.

For instance, could there be two or more standalone goals to which there are separate non-interfering paths?

C. Implementation in 3D

For simplicity of presentation, our algorithm is discussed for the planar setting. Yet, its completeness and near-optimality guarantees hold also for the three-dimensional case with ball robots under the two separation assumptions used in this paper. A main requirement for a correct implementation of the algorithm is the ability to produce shortest paths for a single robot moving amid static obstacles. While this task does not pose particular challenges in the planar case, it becomes extremely demanding in 3D. In particular, finding a shortest path for a polyhedral robot translating amid polyhedral obstacles is known to be NP-Hard [45]. It would be interesting to investigate whether our algorithm can be extended to work with approximate shortest paths.

References

[1] G. Calinescu, A. Dumitrescu, and J. Pach, "Reconfigurations in graphs and grids," *SIAM Journal on Discrete Mathematics*, vol. 22, no. 1, pp. 124–138, 2008.

[2] J. Yu and S. M. LaValle, "Multi-agent path planning and network flow," in *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, 2012, pp. 157–173.

[3] M. Turpin, N. Michael, and V. Kumar, "Concurrent assignment and planning of trajectories for large teams of interchangeable robots," in *International Conference on Robotics and Automation (ICRA)*, 2013.

[4] J. Yu and S. M. LaValle, "Distance optimal formation control on graphs with a tight convergence time guarantee," in *Proceedings IEEE Conference on Decision & Control*, 2012, pp. 4023–4028.

[5] K. Solovey and D. Halperin, "On the hardness of unlabeled multi-robot motion planning," in *Robotics: Science and Systems (RSS)*, 2015, these proceedings.

[6] A. Adler, M. de Berg, D. Halperin, and K. Solovey, "Efficient multi-robot motion planning for unlabeled discs in simple polygons," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.

[7] M. Turpin, N. Michael, and V. Kumar, "Trajectory planning and assignment in multirobot systems," in *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, 2012, pp. 175–190.

[8] I. Karamouzas, R. Geraerts, and A. F. van der Stappen, "Space-time group motion planning," in *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, 2012, pp. 227–243.

[9] A. Krontiris, R. Luna, and K. E. Bekris, "From feasibility tests to path planners for multi-agent pathfinding," in *Symposium on Combinatorial Search, (SOCS), Leavenworth, Washington, USA*, 2013.

[10] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of aerial robots," in *Robotics: Science and Systems*, 2013.

[11] J. Yu and S. M. LaValle, "Planning optimal paths for multiple robots on graphs," in *International Conference on Robotics and Automation (ICRA)*, 2013, pp. 3612–3617.

[12] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI Magazine*, vol. 29, no. 1, pp. 9–19, 2008.

[13] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem"," *International Journal of Robotics Research*, vol. 3, no. 4, pp. 76–88, 1984.

[14] P. G. Spirakis and C.-K. Yap, "Strong NP-hardness of moving many discs," *Information Processing Letters*, vol. 19, no. 1, pp. 55–59, 1984.

[15] J. T. Schwartz and M. Sharir, "On the piano movers' problem: III. coordinating the motion of several independent bodies: the special case of circular bodies moving amidst polygonal barriers," *International Journal of Robotics Research*, vol. 2, no. 3, pp. 46–75, 1983.

[16] M. A. Erdmann and T. Lozano-Pérez, "On multiple moving objects," in *International Conference on Robotics and Automation (ICRA)*, 1986, pp. 1419–1424.

[17] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics & Automation*, vol. 14, no. 6, pp. 912–925, 1998.

[18] J. Peng and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths," in *Algorithmic Foundations of Robotics V*, J.-D. Boissonat, J. Burdick, K. Goldberg, and S. Hutchinson, Eds. Berlin: Springer-Verlag, 2002, pp. 221–237.

[19] J. van den Berg and M. H. Overmars, "Prioritized motion planning for multiple robots," in *International Conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 430 – 435.

[20] R. Ghrist, J. M. O'Kane, and S. M. LaValle, "Computing Pareto Optimal Coordinations on Roadmaps," *International Journal of Robotics Research*, vol. 24, no. 11, pp. 997–1010, 2005.

[21] J. van den Berg, J. Snoeyink, M. Lin, and D. Manocha, "Centralized path planning for multiple robots: Optimal decoupling into sequential plans," in *Robotics: Science and Systems*, 2009.

[22] P. A. O'Donnell and T. Lozano-Pérez, "Deadlock-free and collision-free coordination of two robot manipulators," in *International Conference on Robotics and Automation (ICRA)*, 1989, pp. 484–489.

[23] P. Švestka and M. H. Overmars, "Coordinated path planning for multiple robots," *Robotics and Autonomous Systems*, vol. 23, no. 3, pp. 125–152, 1998.

[24] B. Aronov, M. de Berg, A. F. van der Stappen, P. Švestka, and J. Vleugels, "Motion planning for multiple robots," *Discrete & Computational Geometry*, vol. 22, no. 4, pp. 505–525, 1999.

[25] S. Kloder and S. Hutchinson, "Path planning for permutation-invariant multirobot formations," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 650–665, 2006.

[26] O. Salzman, M. Hemmer, and D. Halperin, "On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2012, pp. 313–329.

[27] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.

[28] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artif. Intell.*, vol. 219, pp. 1–24, 2015.

[29] E. J. Griffith and S. Akella, "Coordinating multiple droplets in planar array digital microfluidic systems," *International Journal of Robotics Research*, vol. 24, no. 11, pp. 933–949, 2005.

[30] D. Kornhauser, G. Miller, and P. Spirakis, "Coordinating pebble motion on graphs, the diameter of permutation groups, and applications," in *Foundations of Computer Science (FOCS)*. IEEE Computer Society, 1984, pp. 241–250.

[31] V. Auletta, A. Monti, M. Parente, and P. Persiano, "A linear time algorithm for the feasibility of pebble motion on trees," in *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, 1996, pp. 259–270.

[32] G. Goraly and R. Hassin, "Multi-color pebble motion on graphs," *Algorithmica*, vol. 58, no. 3, pp. 610–636, 2010.

[33] J. Yu, "A linear time algorithm for the feasibility of pebble motion on graphs," *CoRR*, vol. abs/1301.2342, 2013.

[34] D. Kornhauser, "Coordinating pebble motion on graphs, the diameter of permutation groups, and applications," M.Sc. Thesis, Department of Electrical Engineering and Computer Scienec, Massachusetts Institute of Technology, 1984.

[35] R. Luna and K. E. Bekris, “An efficient and complete approach for cooperative path-finding,” in *Conference on Artificial Intelligence, San Francisco, California, USA*, 2011.

[36] M. Katsev, J. Yu, and S. M. LaValle, “Efficient formation path planning on large graphs,” in *2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany*, 2013, pp. 3606–3611.

[37] K. Solovey and D. Halperin, “*k*-Color multi-robot motion planning,” *International Journal of Robotic Research*, vol. 33, no. 1, pp. 82–97, 2014.

[38] E. L. Lawler, *Combinatorial optimization: networks and matroids*. Courier Dover Publications, 1976.

[39] K. Kedem, R. Livne, J. Pach, and M. Sharir, “On the union of jordan regions and collision-free translational motion amidst polygonal obstacles,” *Discrete & Computational Geometry*, vol. 1, pp. 59–70, 1986.

[40] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed. Springer-Verlag, 2008.

[41] “CGAL, Computational Geometry Algorithms Library,” http://www.cgal.org.

[42] E. Fogel, D. Halperin, and R. Wein, *CGAL Arrangements and Their Applications - A Step-by-Step Guide*, ser. Geometry and computing. Springer, 2012, vol. 7.

[43] “Implementation of the Kuhn-Munkres algorithm,” https://github.com/saebyn/munkres-cpp.

[44] L. Kettner, K. Mehlhorn, S. Pion, S. Schirra, and C. Yap, “Classroom examples of robustness problems in geometric computations,” *Comput. Geom.*, vol. 40, no. 1, pp. 61–78, 2008.

[45] J. F. Canny and J. H. Reif, “New lower bound techniques for robot motion planning problems,” in *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA*, 1987, pp. 49–60.